This listing of claims will replace all prior versions, and listings, of claims in the application:

LISTING OF CLAIMS

1. (Currently Amended) A method of generating a document, the method comprising:

creating a transaction data set;

establishing ~~a software architecture for~~ a set of computer-processable rules in accordance with a rules markup language~~, where~~;

configuring each rule in the set of computer-processable rules ~~is configured~~ to be embedded in one or more computer-processable ~~documents~~ dynamic document structures and to determine the content to be included in at least one instance of a document generated from one of the one or more computer-processable dynamic document structures;[[,]]

creating a computer-implemented database;

storing each rule in the set of computer-processable rules in the database;

storing content in the database;

associating the stored content with one or more rules from the set of computer-processable rules in the database;

~~the documents consisting of a plurality of components, the set of rules defining content to be included in each instance of the one or more computer-processable documents;~~

~~creating a~~ configuring each dynamic document structure to have a tree-architecture, to resolve ~~that resolves~~ to one or more instances of a document, and ~~that is configured~~ to include document content including one or more embedded rules~~ based on the software architecture for the set of rules~~; and

resolving, with a computer processor and in accordance with the transaction data set, ~~the~~ at least one dynamic document structure by executing the one or more embedded rules

~~included in the document content~~ to create a specific instance of a document in a static form.

2. (Currently Amended) A method as claimed in claim 1, wherein establishing ~~a software architecture for a set of rules~~ a set of computer-processable rules in accordance with a rules markup language includes creating a schema having a conditions element.

3. (Currently Amended) A method as claimed in claim 1, wherein establishing ~~a software architecture for a set of rules~~ a set of computer-processable rules in accordance with a rules markup language includes creating a schema having a choose element.

4. (Currently Amended) A method as claimed in claim 1, wherein establishing ~~a software architecture for a set of rules~~ a set of computer-processable rules in accordance with a rules markup language includes creating a schema having an iterators element.

5. (Currently Amended) A method as claimed in claim 1, wherein establishing ~~a software architecture for a set of rules~~ a set of computer-processable rules in accordance with a rules markup language includes creating a schema having a functions element.

6. (Currently Amended) A method as claimed in claim 1, wherein establishing ~~a software architecture for a set of rules~~ a set of computer-processable rules in accordance with a rules markup language includes creating a schema having a conditions element, a choose element, an iterators element, and a functions element.

7. (Currently Amended) A method as claimed in claim 1, wherein establishing ~~a software architecture for a set of rules~~ a set of computer-processable rules in accordance with a rules markup language includes creating a schema having an external interface element that is configured to be resolved into a value.

8. (Previously Presented) A method as claimed in claim 7, wherein the value is chosen from a group that includes a set, an XML DOM node, and an XML DOM node list.

9. (Previously Presented) A method as claimed in claim 7, wherein the external data interface element is configured to have an entity reference attribute.

10. (Previously Presented) A method as claimed in claim 7, wherein the external data interface element is configured to have a return type attribute.

11. (Currently Amended) A method as claimed in claim 1, wherein establishing ~~a software architecture for a set of rules~~ a set of computer-processable rules in accordance with a rules markup language includes creating a schema having an internal interface element and an external interface element.

12. (Previously Presented) A method as claimed in claim 1, further comprising creating a static document structure that can be resolved into one or more instances of a document that includes at least some content that is determined before and some content that is unchanged during and after a resolution process.

13. (Currently Amended) A method as claimed in claim 1, further comprising providing a data set configured to be processable by one or more rules built on the architecture for a set of computer-processable rules.

14. (Currently Amended) A method of generating a document, the method comprising:

creating a transaction data set;

establishing ~~a software architecture for a set of rules~~ a set of computer-processable rules in accordance with a rules markup language, wherein the set of computer-processable rules are configured to be embedded in documents by creating a schema having a conditions element, a choose element, an iterators element, a functions element, and an external interface element that is configured to be resolved into a value;

configuring each rule in the set of computer-processable rules to be embedded in one or more computer processable dynamic document structures and to define ~~defining~~ content to be included in at least one instance of a document generated from one of the one or more computer-processable dynamic document structures; ~~in the documents;~~

creating a computer-implemented database;

storing each rule in the set of computer-processable rules in the database;

storing content in the database;

associating the stored content with one or more rules from the set of computer-processable rules in the database;

~~creating a~~ configuring each dynamic document structure ~~that can resolve to one or more instances of a document using the set of rules, the dynamic document structure including~~ to have a tree-architecture, to resolve to one or more instances of a document, and to include document content that includes one or more embedded rules; and

resolving at least one ~~the~~ dynamic document structure, with a computer processor and in accordance with the transaction data set, by executing the one or more embedded rules embedded in the document content to create a specific instance of a document in a static form.

15. (Currently Amended) A method of generating a document, the method comprising:

creating a transaction data set;

establishing ~~a software architecture for a set of rules~~ a set of computer-processable rules in accordance with a rules markup language, wherein the set of computer-processable rules are configured to be embedded in documents, the set of computer-processable rules including a conditions element, a choose element, an iterators element, and a functions element and an external interface element, the set of computer-processable rules defining content to be included in the documents;

creating a computer-implemented database;

storing each rule in the set of computer-processable rules in the database;

storing content in the database;

associating the stored content in the database with one or more rules from the set of computer-processable rules in the database;

creating a dynamic document structure to have a tree architecture and that can resolve to one or more instances of a document using the set of computer-processable rules, the dynamic document structure including document content that includes one or more embedded rules based on the set of computer-processable rules;

creating a static document structure that can be resolved into one or more instances of a document that includes at least some content that is determined before and some content that is determined during ~~and after~~ a resolution process; and

resolving, with a computer processor and in accordance with the transaction data set, the static document structure by executing the one or more rules embedded in the document content to create a specific instance of a document in a static form.

16. (Currently Amended) A method of assembling a document from a group of components, the method comprising:

creating a transaction data set;

retrieving one or more cross-referenced document components from a data base based on the transaction data set, the one or more document components configured to include document content and one or more embedded rules, the one or more embedded rules defining content to be included in documents;

processing the one or more cross-referenced document components in a processor to generate a tree having a root node;

processing the tree beginning at the root node; ~~and~~

when a rule is encountered, evaluating the rule and replacing it with a value; and

creating a specific instance of a document in a static form based on the transaction data set and the processing of the one or more cross-referenced document components.

17. (Currently Amended) A method as claimed in claim 16, further comprising establishing an architecture for a set of <u>computer-processable</u> rules.

18. (Currently Amended) A method as claimed in claim 17, wherein establishing an architecture for a set of <u>computer-processable</u> rules includes creating a schema having a conditions element.

19. (Currently Amended) A method as claimed in claim 17, wherein establishing an architecture for a set of <u>computer-processable</u> rules includes creating a schema having a choose element.

20. (Currently Amended) A method as claimed in claim 17, wherein establishing an architecture for a set of <u>computer-processable</u> rules includes creating a schema having an iterators element.

21. (Currently Amended) A method as claimed in claim 17, wherein establishing an architecture for a set of <u>computer-processable</u> rules includes creating a schema having a functions element.

22. (Currently Amended) A method as claimed in claim 17, wherein establishing an architecture for a set of <u>computer-processable</u> rules includes creating a schema having a conditions element, a choose element, an iterators element, and a functions element.

23. (Currently Amended) A method as claimed in claim 17, wherein establishing an architecture for a set of <u>computer-processable</u> rules includes creating a schema having an external interface element that is configured to be resolved into a value.

24. (Previously Presented) A method as claimed in claim 23, wherein the value is chosen from a group that includes a set, an XML DOM node, and an XML DOM node list.

25. (Previously Presented) A method as claimed in claim 23, wherein the external data interface element is configured to have an entity reference attribute.

26. (Previously Presented) A method as claimed in claim 23, wherein the external

data interface element is configured to have a return type attribute.

27. (Currently Amended) A method as claimed in claim 17, wherein establishing an architecture for a set of computer-processable rules includes creating a schema having an internal interface element and an external interface element.

28. (Currently Amended) A method of assembling a data structure from a group of components, the method comprising:

creating a transaction data set;

retrieving one or more cross-referenced data structure components from a database based on the transaction data set, the one or more data structure components configured to include document content and one or more embedded rules, the one or more embedded rules defining content to be included in documents;

processing the one or more cross-referenced data structure components in a processor to generate a tree having a root node;

processing the tree beginning at the root node; ~~and~~

when a rule is encountered, evaluating the rule and replacing it with a value; and

creating a specific instance of a document in a static form based on the transaction data set and the processing of the one or more cross-referenced document components.

29. (Currently Amended) A method as claimed in claim 28, further comprising establishing an architecture for a set of computer-processable rules.

30. (Previously Presented) A method as claimed in claim 28, further comprising establishing a list of data structures and performing each of the steps in claim 28 for each of the data structures.